M1 INFO IHM - Introduction à Unity

Hakim Ferrier-Belhaouari

9 février 2024

Les moteurs de jeu vidéo

Inclus des moteurs

- graphique 3D
- graphique 2D
- physique
- audio (2D/3D)
- réseau
- intelligence artificielle (pathfinding, ...)
- animation
- ...

En bref

- C'est compliqué car mêle beaucoup de spécialité.
- Au final, nous avons une boucle perpétuelle de mise à jour de l'image produite et de l'animation d'une *frame*.

Unity

C'est quoi ?

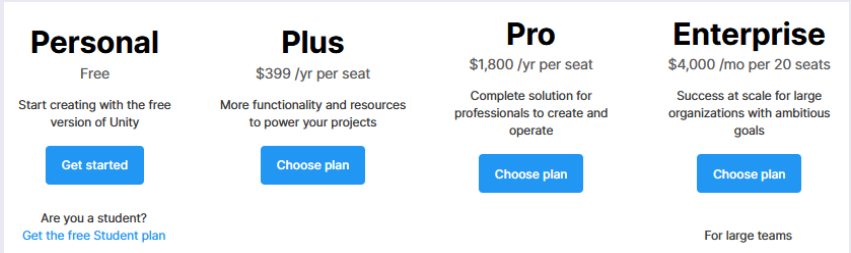
Unity est un **moteur de jeu multiplateformes** pour les jeux vidéos sous la forme d'un **environnement de développement intégré**.

Historique

- 2000 : Création des prémisses d'Unity par 3 danois : volonté de mettre en avant leur expertise.
- 2004 : Fondation de Unity Technologies à San Francisco.
- 2005 : première version de Unity (double licences).
- 2009 : ajout de Direct3D.
- 2010 : création de l'Asset Store (achat d'objet ou autre).
- 2011 : achat de Mecanim : ajout d'animations simples directement dans Unity + ajout de DirectX 11.
- 2013 : ajout d'optimisation des jeux 2D pour répondre à la demande.
- 2015 : intégration de NVIDIA PhysX.
- 2017 : intégration un rendu graphique temps réel.

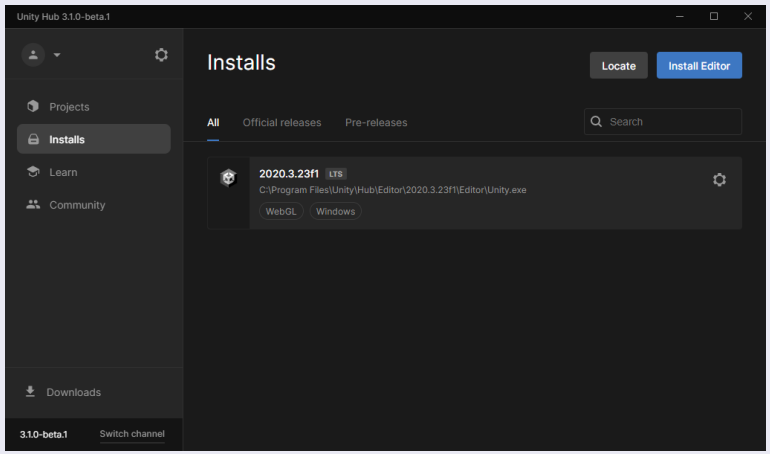
Versions

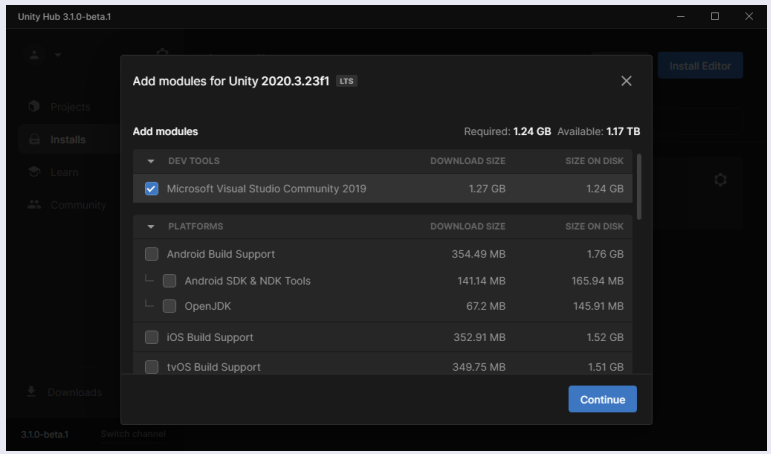
Succès pour des raisons économiques



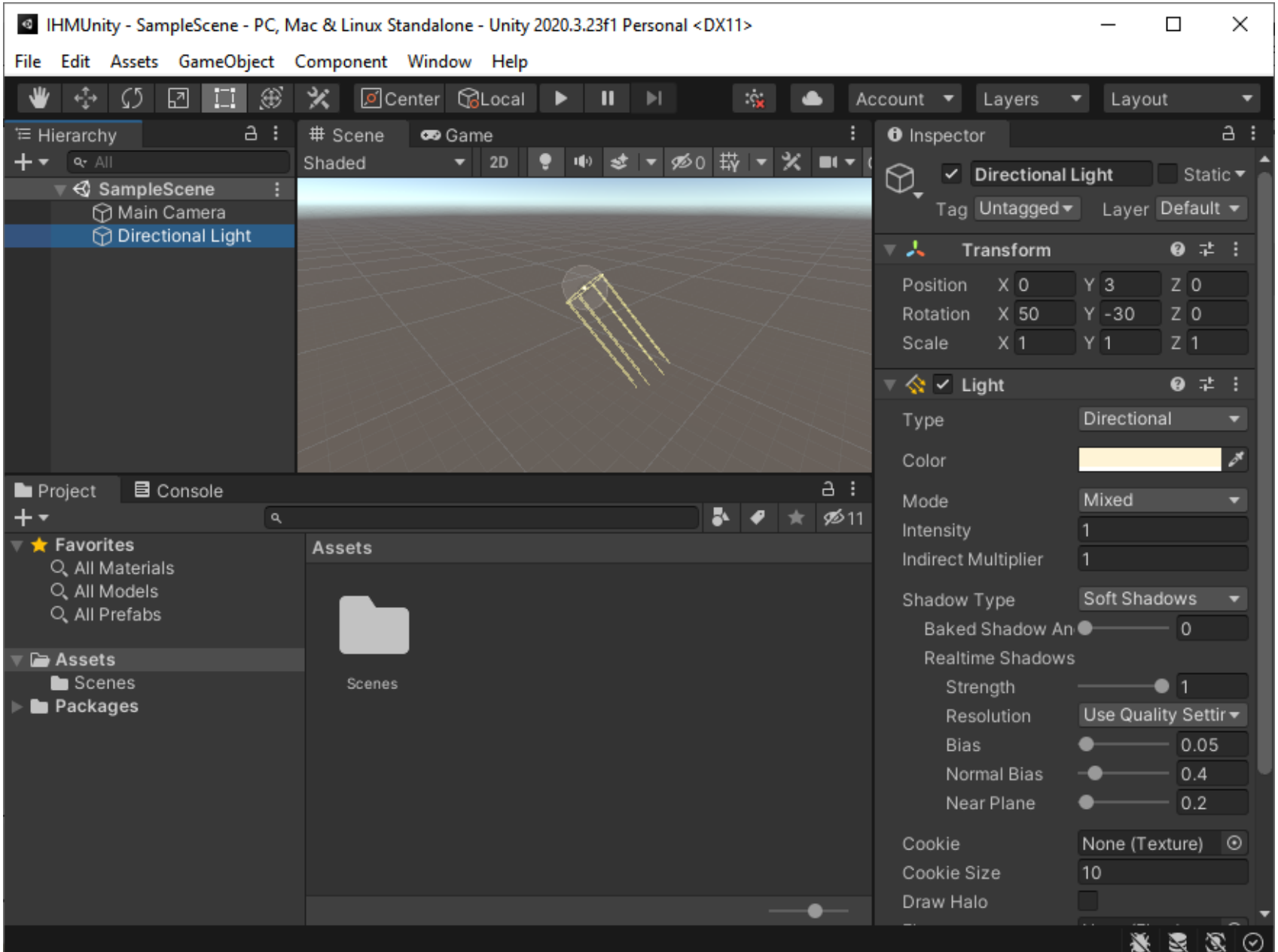
La version personnelle est amplement suffisante pour ce cours, car fonctionne sous les plateformes classiques.

Mécanisme de HUB (version cours : 2022.3 LTS)





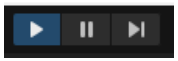
Lancement d'Unity



Lancement d'Unity

Les onglets sont importants et en voici les principaux :

1. Hierarchy : donne une vue sous forme d'arbre de la zone 3D et des objets présents dans votre `scène`(= correspond à un niveau de votre jeu) ou vos scènes.
2. Scene : présente la vue 3D de la scène en cours.
3. Game : vu de votre application quand vous l'exécutez en direct.
4. Inspector : donne l'ensemble des propriétés de votre sélection.
5. Project : permet d'explorer l'entièreté de votre projet (dont les Assets)
6. Console : donne tous les messages de log de Unity.

La barre de lancement 

1. lance l'execution de votre application directement. Et arrête son exécution aussi.
2. met en pause l'exécution (pratique pour déboguer).
3. et le bouton step toujours pour le débogage.

Demo1

Objectifs

- Créez un projet 3D nommé `IHMUnity_Demo1`.
- Ajoutez des objets primitives (1 cube et 1 sphère) dans la vue 3D.
- Regardez leurs propriétés et repérez les similitudes et différences.
- Lancez l'exécution de la vue.
- Pourquoi rien ne se passe ?
- Modifier la position du cube en direct sans arrêter la simulation.
- Stopper la simulation.
- Que remarquez vous sur les positions ?

La classe GameObject

Definition

La classe **GameObject** est la classe de base de tous les objets présents dans une scène. Leur spécificité est la notion de composant (**Component**).

Exemple : Transform

Le composant Transform gère le position de l'objet dans la scène.

Exemple : MeshFilter

Le composant MeshFilter gère le maillage de l'objet.

Les composants

Les composants représentent le cœur de la programmation en Unity. Ainsi, pour un objet il suffit de lui ajouter le bon composant pour lui modifier/influencer/ajouter un comportement ou une mécanique. Lorsqu'il n'y a pas de composant adéquat à votre besoin il nous suffit de le programmer !

Demo2

Objectifs

Sur le projet précédent :

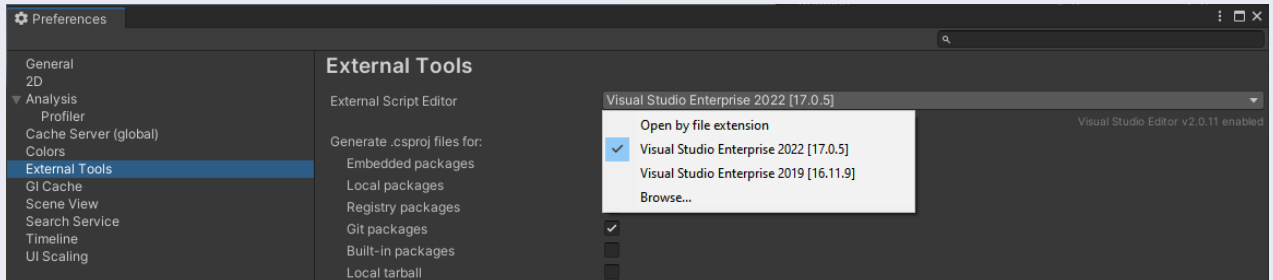
- Ajoutez un composant système de particule au cube.
- Amusez vous pour voir les modifications induits en simulation.
- Stoppez tout.
- Ajoutez un objet 3D Quad, plane ou cube.
- Ajoutez un composant vidéo.
- Lancez la simulation pour voir l'animation de la vidéo sur l'objet.
- Stopper la simulation.
- Que remarquez vous sur les positions ?

Composant script

Prélude

Comme nous allons devoir coder, donc réflexes de développeur :

- Préparation de la documentation officielle : `https://docs.unity3d.com/`. Cette dernière se décompose en un manuel d'utilisation (guide) et d'une référence technique (Scripting API).
- La référence technique montre 2 espaces de noms distincts :
  1. UnityEngine : qui contient l'ensemble des fonctionnalités accessible à l'exécution.
  2. UnityEditor : qui contient l'ensemble des fonctionnalités dans le mode éditeur (celui qu'il faut éviter car pas exploitable une fois déployer sur notre plateforme cible, mais bien pratique pour le débogage).
- La configuration de son environnement pour développer en C# : Edit > Preferences > External Tools



Composant script

Attention !

Tous les composants scripts héritent de MonoBehaviour ! Elle possède pleins de fonction en accord avec son cycle de vie :

- Awake : lorsque le script est chargé pour la première fois.
- Start : lorsque le script est lancé pour la première fois.
- Update : callback appelé à chaque frame (plusieurs variantes).
- (détail : `https://docs.unity3d.com/Manual/ExecutionOrder.html`)

Paramétrage de script

Les variables publiques sont reconnaissables par l'éditeur et manipulable directement dans l'éditeur.

Demo3

Objectifs

- Dans une scène, ajoutez un cube/sphère.
- Ajoutez un nouveau script composant nommé : JeTourne.
- Editez le script pour qu'il affiche un message de débogage pour certaines fonctions du cycle de vie et qu'il réalise une rotation autour de lui.
- Dans le script faite une rotation de 5° autour de y par frame. Observez.
- Modifiez pour que cela se fasse dans `FixedUpdate` (avec `Time.fixedDeltaTime`). Observez.
- Réalisez un schéma du système Soleil-Terre-Lune qui tourne autour d'eux et entraine l'objet gravitant avec eux d'un point de vue hiérarchique (aucune importance de la véracité physique).
- Modifiez le script pour avoir un attribut publique qui correspond à la vitesse de rotation.

Les interfaces

Attention !

Il existe plusieurs kits d'interface graphiques en Unity :

Type of UI	UI Toolkit	Unity UI(uGUI)	IMGUI	Notes
Runtime (debug)	✓*	✓	✓	This refers to temporary runtime UI used for debug purposes.
Runtime (in-game)	✓*	✓	Not Recommended	For performance reasons, Unity does not recommend IMGUI for in-game runtime UI.
Unity Editor	✓	✗	✓	You cannot use Unity UI to make UI for the Unity Editor.

** Requires the UI Toolkit package, currently in preview.*

Dans ce cours, nous ferons que du `Unity UI` qui suit les mécaniques des `GameObject`. Cependant, il est impossible d'étendre l'éditeur.

La zone d'interface se compose

Lorsque vous réalisez une interface il est OBLIGATOIRE d'avoir les éléments suivants (Unity peut vous les créer si besoin) :

- Canvas : la zone de dessin qui est la base de toute interface (cf démo)
- EventSystem : un objet faisant le pont avec les entrées (clavier, souris, ...).

Les interfaces

Dans ce cours

Nous construirons nos widgets graphiques par composition d'élements existant et de script pour compenser des manques.

Spécificité des widgets graphiques

Les widgets graphiques n'ont pas de `Transform` qui gère la position dans la scène, mais un `RectTransform` qui s'occupe du placement dans le canevas. Le paramétrage est différent.

Les modes de rendu du Canvas

Un canevas a 3 modes de rendu :

- Screen space - Overlay : qui est mode intuitif 2D comme on connait.
- Screen space - Camera : qui subit les modifications de la caméra (comme la perspective)
- World Space : qui place le canevas dans l'environnement 3D.

Demo

Objectifs

- Dans la scène précédente (Soleil-Terre-Lune).
- Ajoutez bouton comme interface.
- Ajoutez dans le script JeTourne une fonction toggle qui active et désactive la rotation.
- Lié le bouton ajouter à la fonction précédente pour contrôler l'animation sur un objet.
- Vous pouvez ajouter d'autre bouton pour contrôler les rotations des autres astres.

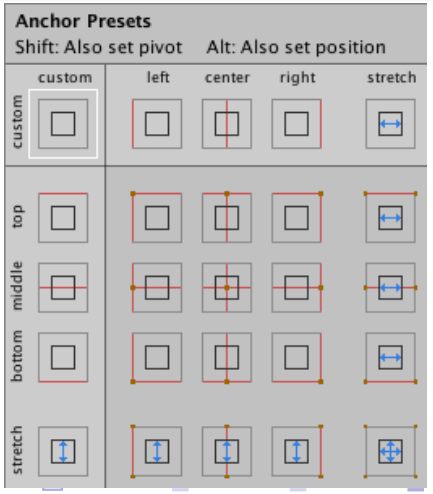
La mise en page

Attention !

Le canevas a toujours une taille précise (dans les paramètres de l'application) et mise à l'échelle. Elle sera donc toujours comme vous la voyez dans l'éditeur !
La mise en page des widgets n'est pas aussi intuitive que les autres outils, en raison de la consommation des replacements.

Les ancres

Tous les widgets ont des ancres qui permettent de fixer leur position au sein de leur parent.



En Vrac

- Notion de Préfabs et création dynamique.
- Mécanisme d'export.
- Prudence Git et soumission de vos projets.
- Mécanique des inputs (attention refonte dans Unity).
- Retrouver un objet ? (En TP)
- Faire plusieurs scènes pour expérimenter.